Constraint Method-Based Evolutionary Algorithm (CMEA) for Multiobjective Optimization

S.Ranji Ranjithan[1], S. Kishan Chetan[2], Harish K Dakshina[3]

[1] Department of Civil Engineering, Campus Box 7908,
North Carolina State University, Raleigh, NC 27695, USA
ranji@eos.ncsu.edu
[2] Siebel Systems, 6001 Shellmound, Emeryville, CA 94608, USA
kchetan@siebel.com
[3] Department of Civil Engineering, Campus Box 7908,
North Carolina State University, Raleigh, NC 27695, USA
hkdakshi@unity.ncsu.edu

Abstract. Evolutionary algorithms are becoming increasingly valuable in solving large-scale, realistic engineering multiobjective optimization (MO) problems, which typically require consideration of conflicting and competing design issues. The new procedure, Constraint Method-Based Evolutionary Algorithm (CMEA), presented in this paper is based upon underlying concepts in the constraint method described in the mathematical programming literature. Pareto optimality is achieved implicitly via a constraint approach, and convergence is enhanced by using beneficial seeding of the initial population. CMEA is evaluated by solving two test problems reported in the multiobjective evolutionary algorithm (MOEA) literature. Performance comparisons based on quantitative metrics for accuracy, coverage, and spread are presented. CMEA is relatively simple to implement and incorporate into existing implementations of evolutionary algorithm-based optimization procedures.

1 Introduction

Tradeoff information in the form of a noninferior, or Pareto optimal set, of solutions is important in considering competing design objectives when making decisions associated with most engineering problems. The use of standard mathematical programming-based techniques, such as constraint method and weighting method (Cohon, 1978) for generating noninferior sets, in many practical engineering problems is limited because the models describing the engineering processes seldom fit into the restrictive function forms required by these mathematical programming techniques. As evolutionary algorithms offer a relatively more flexible way to analyze and solve realistic engineering design problems, their use in multi criterion decision making is becoming increasingly important. An array of multiobjective evolutionary algorithms (MOEAs) has been reported since the early eighties. Detailed summaries of the state-of-the-art in MOEA were discussed recently by Coello (1999a) and Van Veldhuizen and Lamont (2000), and are represented in the special issue of *Evolutionary Computation* (Vol. 8, No. 2, Summer 2000) on multi criterion optimization (also see Coello (1999b) for an archive of bibliography).

The procedure presented in this paper integrates the underlying concepts of traditional multiobjective optimization (MO) techniques into an evolutionary algorithm framework. In the conventional constraint method for MO, one objective is optimized while constraining all other objectives to target levels to identify a noninferior, or Pareto optimal, solution. This is applied iteratively to generate the noninferior set. Solving a number of independent single objective optimization models to obtain the noninferior set, however, renders these MO methods less attractive if obtaining each noninferior solution is computationally intensive. This issue is addressed in some traditional mathematical programming approaches (e.g., linear programming, nonlinear programming) by seeding a search for a new noninferior solution with a previously generated adjacent noninferior solution (i.e., prior basis). Since, for some class of problems, noninferior solutions adjacent in the decision space map to adjacent points in the objective space, this seeding approach reduces the number of steps in the search for each noninferior solution.

This paper describes the development of the Constraint Method-Based Evolutionary Algorithm (CMEA), which explicitly uses this adjacency mapping property to its advantage within an MOEA framework. The underlying simple concepts in CMEA allow it to be applied easily within existing implementations of evolutionary algorithms for single objective optimization. First, the general representation of a standard MO problem and definitions of Pareto optimality are presented, followed by a brief overview of existing MOEAs and their applications. The details of CMEA are then described, followed by its application to a couple of published test problems. An extensive performance comparison of CMEA and four commonly accepted MOEAs in solving one of the published test problems is presented. The noninferior solutions are also compared to those obtained via mathematical programming techniques. Finally, concluding remarks are made with a brief discussion of CMEA's strengths and weaknesses.

2 Background

2.1 A Standard Multiobjective Optimization Problem

A general multiobjective optimization problem consisting of k competing objectives and m constraints defined as functions of decision variable set $\boldsymbol{x}$ can be represented as follows:

$$\textit{Maximize} \quad \mathbf{Z}(\boldsymbol{x}) = \{Z_l(\boldsymbol{x}): l=1,2, \ldots, \mathrm{k}\} \tag{1}$$

$$\textit{Subject to} \quad g_i(\boldsymbol{x}) \leq 0 \quad \forall i = 1,2,\ldots,m \tag{2}$$

$$\boldsymbol{x} \in \boldsymbol{X} \tag{3}$$

where $\boldsymbol{x} = \{x_j : j = 1,2,\ldots..,n\}$ represents the decision vector, x_j is the j^{th} decision variable, $\boldsymbol{X}$ represents the decision space, $g_i(\boldsymbol{x})$ is the i^{th} constraint, $\boldsymbol{Z}(\boldsymbol{x})$ is the multiobjective vector, and $Z_l(\boldsymbol{x})$ is the l^{th} objective function.

2.2 Noninferiority

Noninferiority, which is also referred as nondominance or Pareto optimality, of a set of solution is formally defined as follows (Cohon, 1978): a feasible solution to a multiobjective problem is non-inferior if there exists no other feasible solution that will yield an improvement in one objective without causing a degradation in at least one other objective.

Van Veldhuizen and Lamont (2000) and Zitzler et al. (2000) provide a more rigorous definitions of this and related MO terminology. Based on the definitions by Van Veldhuizen and Lamont (2000) and notations used in Equations 1-3, the following are defined:

Pareto Dominance: A multiobjective vector $\boldsymbol{u} = (u_1, u_2, \ldots u_k)$ is said to dominate $\boldsymbol{v} = (v_1, v_2, \ldots v_k)$ (denoted by $\boldsymbol{u} \succeq \boldsymbol{v}$) if and only if $\boldsymbol{u}$ is partially more than $\boldsymbol{v}$, i.e., $\forall i \in \{1,2,\ldots k\},\ u_i \geq v_i \wedge \exists i \in \{1,2,\ldots k\} : u_i > v_i$.

Pareto Optimality: A solution $x \in X$ is said to be Pareto optimal with respect to X if and only if there exists no $x' \in X$ for which $v = Z(x')$ dominates $u = Z(x)$.

Pareto Optimal Set: For a given multiobjective problem $\boldsymbol{Z(x)}$, the Pareto optimal set $\boldsymbol{P}^*$ is a set consisting of Pareto optimal solutions. $\boldsymbol{P}^*$ is a subset of all the possible solutions in $\boldsymbol{X}$. Mathematically, $\boldsymbol{P}^*$ is defined as follows:

$$\boldsymbol{P}^* := \{\boldsymbol{x} \in \boldsymbol{X} \mid \neg\exists\, \boldsymbol{x}' \in \boldsymbol{X} : \boldsymbol{Z(x')} \succeq \boldsymbol{Z(x)}\} \qquad (4)$$

Pareto Front: The Pareto front, $\boldsymbol{PF}^*$ is the set that contains the evaluated objective vectors of $\boldsymbol{P}^*$. Mathematically $\boldsymbol{PF}^*$ is defined as:

$$\boldsymbol{PF}^* := \{\ \boldsymbol{u} = \boldsymbol{Z(x)} \mid \boldsymbol{x} \in \boldsymbol{P}^*\ \} \qquad (5)$$

2.3 Multiobjective Evolutionary Algorithms (MOEAs)

Many procedures have been reported since the pioneering work by Schaffer (1984, 1985) in the area of EAs for MO optimization. Fonesca and Fleming (1993, 1995), Horn (1997), Coello (1999a, 1999b), and Van Veldhuizen and Lamont (2000) present detailed surveys and catalogues of existing MOEAs. Many different bases, e.g., differences in fitness and selection implementations, for higher-level classification of MOEAs are used in these surveys. For instance, Schaffer's (1985) vector evaluated genetic algorithm (VEGA) uses a special single-objective-based preferential selection procedure, the method by Hajela and Lin (1992) uses an aggregated fitness function, and the methods by Horn et al. (1994), Srinivas and Deb (1994), Zitzler and Thiele (1999), and Knowles and Corne (2000) use Pareto-based selection procedures to determine the noninferior set. Further, these techniques can be categorized by special operators they use, such as niching and sharing (e.g., Horn et. al., 1994; Menczer et. al., 2000), restrictive mating (e.g., Loughlin and Ranjithan, 1997), and elitism (e.g., Knowles and Corne, 2000; Zitzler and Thiele, 1999). Loughlin et al. (2000a) recently presented an EA-based approach that addresses problems with multiple objectives where some may not be easily quantified or modeled. Applications discussed by Hajela and Lin (1992), Ritzel et al. (1994), Cieniawski et al. (1995), Jimenez and

Cadenas (1995), Harrell and Ranjithan (1997), Coello et al. (1998), Coello and Christiansen (2000), Loughlin et al. (2000b), Obayashi et al. (2000) are some examples of use of MOEAs in multiobjective optimization of varied realistic engineering problems. VEGA-Vector Evaluated Genetic Algorithm (Schaffer, 1985), NPGA-Niched Pareto Genetic Algorithm (Horn et al., 1994), NSGA-Non-dominated Sorting Genetic Algorithm (Srinivas and Deb, 1994), and SPEA-Strength Pareto Evolutionary Algorithm (Zitzler and Thiele, 1999) are the four selected MOEAs that are used to compare the performance of CMEA presented in this paper.

3 CMEA – Constraint Method-Based Evolutionary Algorithm

Based on the concepts of the mathematical programming-based constraint approach (Cohon, 1978) for generating the noninferior set, CMEA achieves Pareto optimality in an implicit manner by ensuring that the population migrates along the noninferior surface. At each iteration, the population converges to a noninferior solution by solving the following single objective optimization problem:

$$\textit{Maximize} \quad Z_h(\boldsymbol{x}) \tag{6}$$

$$\textit{Subject to} \quad g_i(\boldsymbol{x}) \leq 0 \qquad \forall i = 1,2,\ldots,m \tag{7}$$

$$Z_l(\boldsymbol{x}) \geq \mathrm{Z}_l^t \qquad \forall l = 1,2,\ldots,k;\ l \neq \mathrm{h} \tag{8}$$

$$\boldsymbol{x} \in \boldsymbol{X} \tag{9}$$

where, Z_h is one of the k objectives, and Z_l^t $\{l=1,2,\ldots k;\ l\neq h\}$ is the constraint value for objective l ($\neq h$) corresponding to a noninferior solution. By varying Z_l^t incrementally, the search migrates from one noninferior solution to an adjacent solution, eventually tracing the noninferior surface. A two-objective illustration is shown in Figure 1.

At each intermediate step in which one noninferior solution is obtained, the model given by Equations (6)-(9) is solved. A straightforward implementation of an algorithm that repeats this intermediate step would be similar to iterative execution of a single objective EA, which is not necessarily computationally efficient. Instead, CMEA exploits the basic concept that for some classes of problems, adjacent solutions in the decision space map to adjacent points in the objective space. Its implication is that these decision vectors ($\boldsymbol{x}$s) (that map to adjacent noninferior points in the objective space) have solution features (i.e., values of x_js) that are only marginally different. This enables the beneficial use of the final population corresponding to the current noninferior solution to seed the search of an adjacent noninferior solution. The new search of course would have an updated constraint set (8) to represent an adjacent noninferior point in the objective space. When the new selection pressure manifesting from the updated constraint vector is applied on the previous population, the population quickly migrates to an adjacent noninferior solution. A systematic update of the constraint set (8) thus enables an efficient mechanism for incrementally tracing the noninferior set. This incremental population migration approach significantly reduces the computational burden compared to that required when solving each single objective EA as independent search problems.

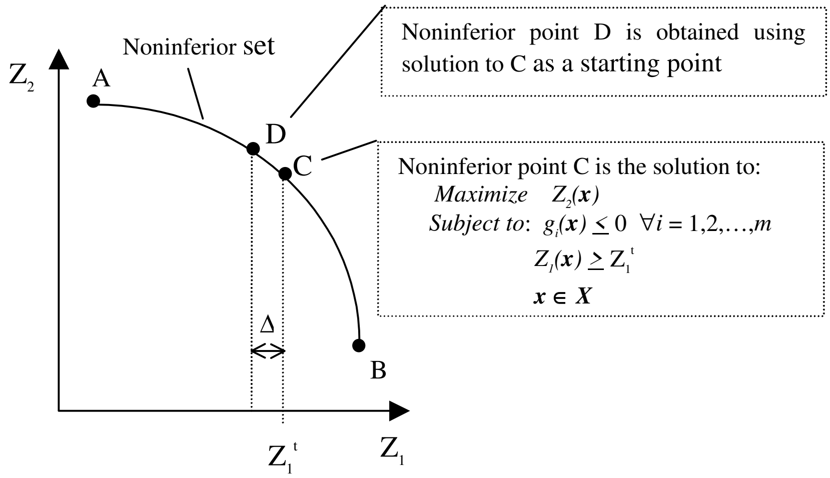

Fig. 1. Illustration of the constraint method using a two-objective example

Using a two objective problem as an illustration (Figure 1), let the current constraint value for objective 1 be Z_1^t. While at noninferior point C, the updated constraint value corresponding to the search for an adjacent noninferior solution D would be (Z_1^t - Δ), where the magnitude of Δ determines the minimum interval desired between adjacent noninferior solutions. For example, smaller values of Δ would result in a finer coverage (or better distribution) of the noninferior set, but would require execution of more intermediate steps, each of which requiring the solution of a single objective EA. At the beginning of the algorithm, the solutions to the two extreme points (A and B in Figure 1) are obtained. Using these solutions, the increment Δ is computed as $(Z_1^B - Z_1^A)/r$, where r is a predefined number of intervals. Then the algorithm starts at extreme point B, and solves the model with the constraint value of (8) set to Z_1^B-Δ. Once the population converges to the adjacent noninferior solution according to some stopping criterion, the best solution is stored. Then the constraint value is incremented adaptively to $Z_1^t \leftarrow Z_1^t$-Δ, and the current population continually undergoes the evolutionary operators and fitness pressure. To introduce higher population diversity at the beginning of each search, the mutation operator is applied in an adaptive manner during each intermediate step, starting with a higher rate and gradually reducing it (e.g., exponential decay) with generations within each step. Thus, at the beginning of each intermediate step the higher mutation rate perturbs the converged population around the previous noninferior point, introducing diversity for the new search.

This iterative process is terminated when the constraint value corresponds to the extreme point A, i.e., when $Z_1^t = Z_1^A$. Two convergence criteria are implemented to determine when to update the constraint value and when to initiate the search for the next noninferior solution. One of the criteria is to check if the number of generations, *generation,* exceeds a maximum value, *maxGenerations*. The other criterion is to track the improvement in the best solution in each intermediate step; convergence is assumed when the best solution does not improve within a certain number (*N*) of successive generations. If either of the above two criteria is satisfied then the constraint value is updated. The key steps of CMEA are shown as a flowchart in Figure 2.

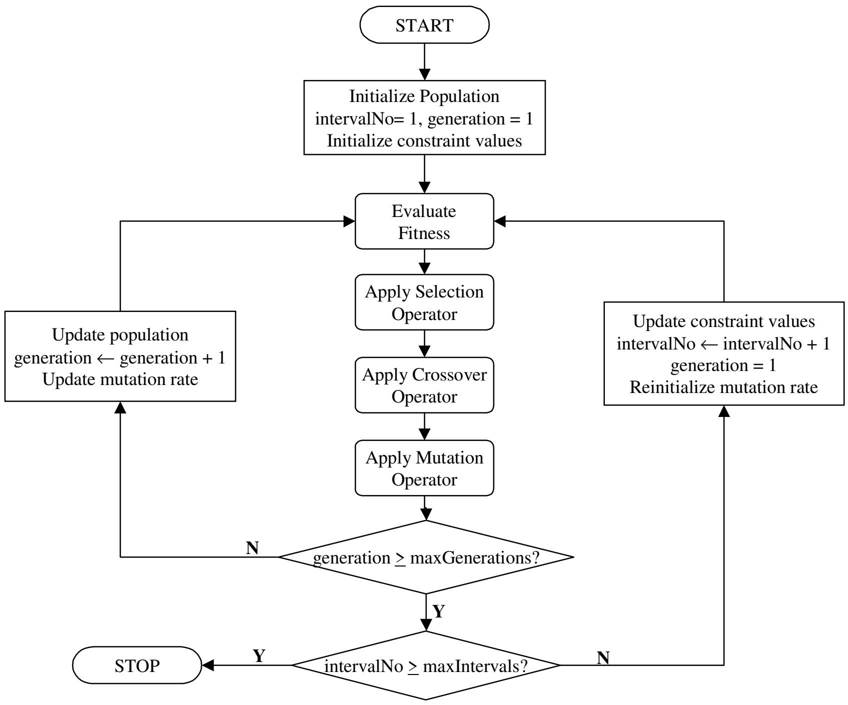

Fig. 2. Flowchart for CMEA – Constraint Method-Based Evolutionary Algorithm

Instead of converging the population to the noninferior set simultaneously, at each intermediate step of CMEA, a point in the noninferior set is identified through a search conducted by the whole population, and the final noninferior set is generated by storing all noninferior solutions found at the intermediate steps. The coverage of the noninferior set is achieved explicitly by traversing the noninferior surface through incremental and systematic updates of the constraint values.

4 Testing and Evaluation of CMEA

CMEA was applied to two test problems of different difficulty and characteristics. The first application uses Schaffer's F2 problem (Schaffer, 1985), which is an unconstrained, nonlinear problem. This is included since most other MOEA methods have been tested against it, providing a common basis for comparison. While this application represents a problem in a continuous search space, the second application, which uses the extended 0/1 multiobjective knapsack problem (Zitzler and Thiele, 1999), represents a problem in a combinatorial search space. This problem is a constrained, binary problem. Performance comparisons of several MOEAs in solving this problem are presented by Zitzler and Thiele (1999), and are used here to compare the performance of CMEA. In addition, a noninferior set was generated using a mathematical programming-based weighting method for the extended 0/1 knapsack problem, which was solved using a binary programming solver CPLEX®.

Several performance criteria are used to evaluate CMEA and to compare it with other approaches: 1) *accuracy*, i.e., how close are the generated noninferior solutions to the best available prediction; 2) *coverage*, i.e., how many different noninferior solutions are generated and how well are they distributed; and 3) *spread*, i.e., what is the maximum range of the noninferior surface covered by the generated solutions. Currently reported as well as newly defined quantitative measures (Chetan, 2000) are used in comparing CMEA with other MOEAs. The robustness of CMEA in solving problems with different characteristics (e.g., real vs. binary variables, constrained vs. unconstrained, continuous vs. combinatorial) is examined, in some limited manner, by applying it to these two different problems. To evaluate the robustness of CMEA in generating the noninferior set and providing good coverage, random trials were performed where the problems were solved repeatedly for different random seeds. A representative solution is used in the comparisons below

4.1 Schaffer's F2 Problem

4.1.1 Description. The F2 problem is defined as follows:

$$\text{Minimize} \qquad Z_1 = x^2 \qquad \textbf{(10)}$$

$$\text{Minimize} \qquad Z_2 = (x-2)^2 \qquad \textbf{(11)}$$

The range for the decision variable x is [-5,7]. The Pareto optimal solutions constitute all x values varying from 0 to 2. The solution $x = 0$ is optimum with respect to Z_1 while the solution $x = 2$ is optimum with respect to Z_2. That is, objective functions Z_1 and Z_2 are in conflict in the range [0,2].

4.1.2 Results. The F2 problem was solved using CMEA with algorithm-specific parameters as shown in Table 1. Results are compared in Figure 3 where the exact solution (obtained analytically using Equations (10) and (11)) for this problem is also shown. Although this is a relatively simple problem, the results indicate that CMEA is very accurate in generating the noninferior set for this problem. Also, it provides good coverage by generating a good distribution of noninferior solutions, and provides a full spread.

Table 1. CMEA parameters and settings for solving the test problems

Problem	**Variable Type**	**CMEA Parameters**			
		No. of intervals	Pop. size	Encoding	Crossover
F2	Real	100	100	32 bit Binary	Uniform
Knapsack	Binary	100	100	Binary	Uniform

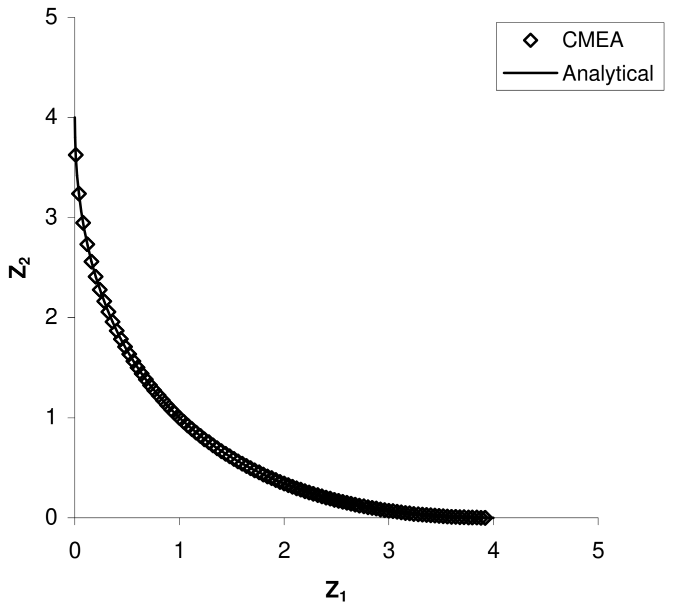

Fig. 3. The true noninferior tradeoff curve and the noninferior set determined by CMEA for Schaffer's F2 problem

4.2 Extended 0/1 Multiobjective Knapsack Problem

4.2.1 Description. Zitzler and Thiele (1999) used in their work a knapsack problem that extends the traditional single objective knapsack problem by incorporating two knapsacks that can be filled by items selected from a larger collection of items. Similar to the traditional knapsack problem, each knapsack has a limited weight capacity with different payoff when each item is included in it. The goal is to allocate a limited set of items to maximize the payoff in each knapsack without violating its weight capacity constraint. This multiobjective problem is defined mathematically as follows:

$$\textit{Maximize} \qquad Z_l(\boldsymbol{x}) = \sum_{j=1}^{n} p_{l,j}\, x_j \qquad \forall l = 1,2,\ldots,k \tag{12}$$

$$\textit{Subject to} \qquad \sum_{j=1}^{n} w_{l,j}\, x_j \leq c_l \qquad \forall l = 1,2,\ldots,k \tag{13}$$

In the formulation, $Z_l(\boldsymbol{x})$ is the total profit associated with knapsack l, $p_{l,j}$ = profit of placing item j in knapsack l, $w_{l,j}$ = weight of item j when placed in knapsack l, c_l = capacity of knapsack l, $\boldsymbol{x} = (x_1, x_2, \ldots\ldots, x_n) \in \{0,1\}^n$ such that $x_j = 1$ if selected and = 0 otherwise, n is the number of available items and k is the number of knapsacks.

This binary MO problem was solved for the cases with two knapsacks (i.e. $k = 2$) and 250 and 500 items. The results reported here correspond to $n = 500$ and $k = 2$. The data for the problems solved were adapted from Zitzler and Thiele (1999).

4.2.2 Results. The extended knapsack problem was solved by CMEA for the parameter setting shown in Table 1. In addition, the noninferior set was generated using the constraint method for this problem by modeling it as a binary linear programming (BLP) model. This was solved using the binary linear programming solver, CPLEX®. In Figure 4, these results are shown along with the results reported by Zitzler and Thiele (1999) for the following MOEAs: VEGA, NPGA, NSGA, and SPEA. To examine the consistency of CMEA in solving this problem, ten trials with different random seeds were conducted, and the results were insensitive to the random seed, indicating robust behavior.

Accuracy of the noninferior solutions generated by CMEA should be compared with respect to the best available noninferior set, as well as with the best estimate obtained by the other MOEAs. The mathematical programming-based estimate of the noninferior set, the best available for this problem, is included in Figure 4 to make the first evaluation. Compared to this, the accuracy of noninferior solutions generated by CMEA and the other MOEAs is relatively poor. The combinatorial nature of the search can be attributed to the weak performance by all EAs. Accuracy of CMEA in comparison to other MOEA results , however, is very good. Noninferior solutions obtained by SPEA, the best performing MOEA according to Zitzler and Thiele (1999), appear to dominate the solutions generated by CMEA in the corresponding noninferior region. The spread or range covered by the CMEA generated solutions, however, is far superior to that attained by all other MOEAs, including SPEA. Further, CMEA is able to provide good coverage by identifying noninferior solutions that are better distributed over a wider range of the noninferior set.

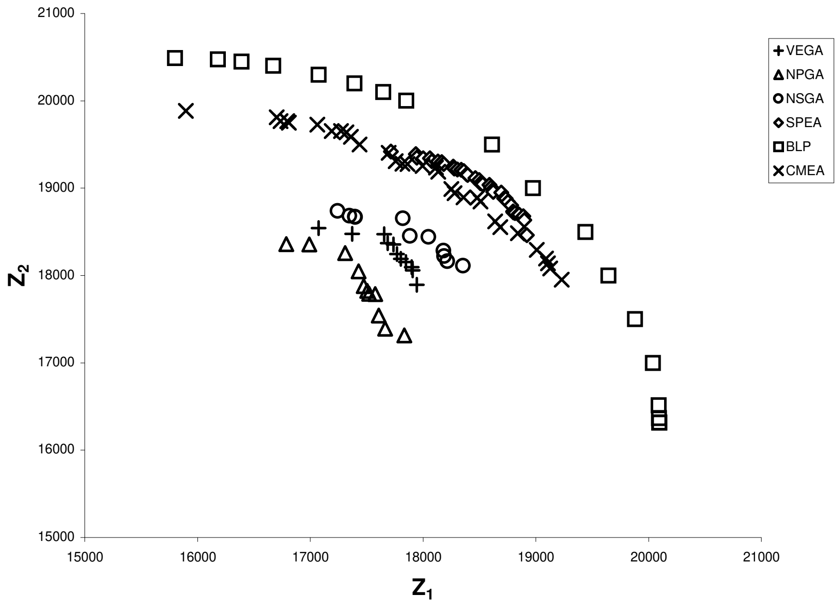

Fig. 4. A comparison of noninferior sets obtained using CMEA, VEGA, NPGA, NSGA, SPEA, and mathematical programming approach (BLP-Binary Linear Programming) for the extended 0/1 multiobjective knapsack problem

5 Performance Metrics and Comparison of MOEAs

To compare the performance of CMEA with that of other MOEAs, the following quantitative measures are used.

- *Accuracy*: The *S* factor used by Zitzler and Thiele (1999) to represent the size of noninferior space covered is used to characterize and compare accuracy. In addition, the approach used by Knowles and Corne (2000) is used to characterize the degree to which a noninferior set outperforms another. An either-or criterion is used to determine if the noninferior set obtained by an MOEA dominates that obtained by another MOEA; the closeness of the two points of intersection are not differentiated statistically.
- *Spread*: Spread is quantified for each objective as the fraction of the maximum possible range of that objective in the noninferior region covered by a noninferior set. A larger value of this metric indicates better spread. As shown in Figure 6, let points A and B refer to the two extreme points, i.e., the single objective optimal solutions for objective 1 and 2, respectively, for a two objective case. The maximum range covered by the noninferior solutions represented by the ordered

set $\boldsymbol{C} = \{C_h, \forall\ h \in \{0,1, \ldots, q\}\}$ is $(Z_1^{Cq} - Z_1^{C1})$ and $(Z_2^{C1} - Z_2^{Cq})$ in Z_1 and Z_2 objective space, respectively. Therefore, the spread metrics in objective space 1 and 2 are defined as $(Z_1^{Cq} - Z_1^{C1})/(Z_1^B - Z_1^A)$ and $(Z_2^{C1} - Z_2^{Cq})/(Z_2^A - Z_2^B)$, respectively.

– *Coverage*: A quantitative measure computed based on the maximum gap in coverage is defined to represent the distribution of the noninferior solutions generated by an MOEA. The Euclidean distance between adjacent noninferior points in the objective space is used to indicate the gap. A smaller value of this metric indicates better distribution of solutions in the noninferior set. This metric is defined separately as *V1* and *V2* to characterize the coverage within the range of noninferior region defined by 1) the extreme points, and 2) the solutions generated by the MOEA, respectively. Using the illustrations shown in Figure 6, *V1* is defined as $Max\ \{d_h, \forall\ h \in \{0,1, \ldots, q\}\}$, and *V2* is defined as $Max\ \{d_h, \forall\ h \in \{1, 2, \ldots, q\text{-}1\}\}$.

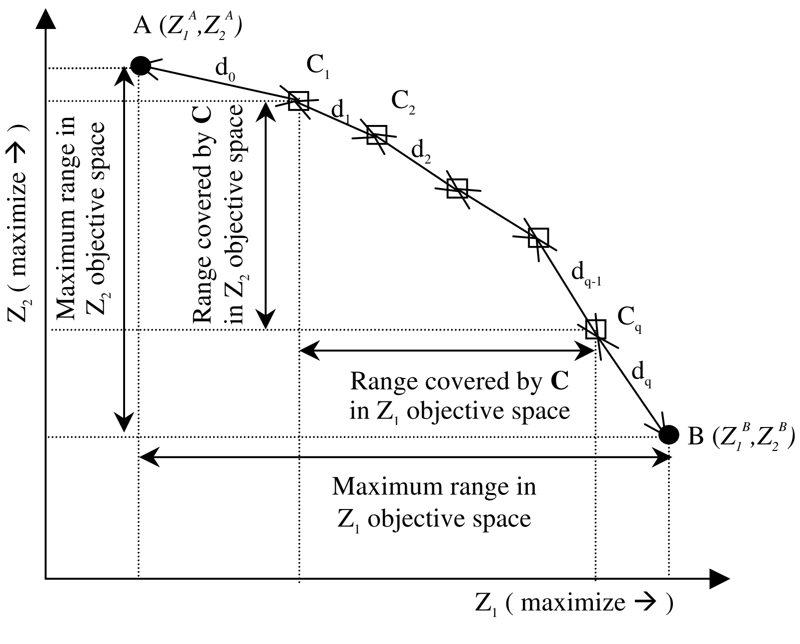

Fig. 5. An example two-objective noninferior tradeoff to illustrate the computation of: 1) *Spread* metric, and 2) *Coverage* metric

A summary of these metrics are compared in Tables 2a-2d for the noninferior solutions generated by all MOEAs shown in Figure 4. These results indicate that overall CMEA performs better than NPGA, NSGA, SPEA, and VEGA with respect to finding nondominated solutions with a good distribution in the noninferior region. This conclusion is specific to the 0/1 extended multiobjective knapsack problem, and similar performance comparisons for other problems are needed to make more general conclusions. Although CMEA provides the best distribution of solutions in the entire noninferior range (based on *V1* metric), SPEA provides a better distribution (based on *V2* metric) within the narrower noninferior range represented by its solutions.

Table 2a. *Accuracy* comparison, based on the *S* factor (Zitzler and Thiele, 1999), of noninferior sets generated by different MOEAs for the extended 0/1 multiobjective knapsack problem. A larger value indicates better performance; the best is shown in bold

MOEA Method	*S* Factor
SPEA	0.89
NSGA	0.79
NPGA	0.83
VEGA	0.81
CMEA	**0.92**

Table 2b. *Accuracy* comparison, based on the metric defined by Knowles and Corne (2000), of CMEA with different MOEAs for the extended 0/1 multiobjective knapsack problem

The MOEAs Compared ($MOEA_1$ vs. $MOEA_2$)	(P_1, P_2): (Percentage number of times $MOEA_1$ outperforms $MOEA_2$, Percentage number of times $MOEA_2$ outperforms $MOEA_1$) Number of Sampling Lines		
	108	**507**	**1083**
(CMEA vs. SPEA)	(95.4, 4.6)	(95.3, 4.7)	(95.2, 4.8)
(CMEA vs. NSGA)	(100, 0)	(100, 0)	(100, 0)
(CMEA vs. NPGA)	(100, 0)	(100, 0)	(100, 0)
(CMEA vs. VEGA)	(100, 0)	(100, 0)	(100, 0)

Table 2c. Comparison of *Spread* of noninferior sets generated by different MOEAs for the extended 0/1 multiobjective knapsack problem. A larger value indicates better performance; the best is shown in bold

MOEA	Spread Metric	
	in Z_1 objective space	**in Z_2 objective space**
SPEA	0.28	0.23
NPGA	0.24	0.25
NSGA	0.26	0.15
VEGA	0.20	0.16
CMEA	**0.78**	**0.47**

Table 2d. Comparison of *Coverage* of noninferior sets generated by different MOEAs for the extended 0/1 multiobjective knapsack problem. A smaller value indicates better performance; the best is shown in bold

MOEA	Coverage Metric	
	V1 **(includes the extreme points for each objective)**	***V2*** **(excludes the extreme points for each objective)**
SPEA	0.118	**0.011**
NPGA	0.122	0.016
NSGA	0.121	0.021
VEGA	0.130	0.015
CMEA	**0.088**	0.04

6 Summary and Conclusions

This paper presents a new MOEA, CMEA-Constrain Method-based Evolutionary Algorithm for solving multiobjective optimization problems. CMEA is founded upon two simple, but powerful concepts borrowed from the mathematical programming literature: 1) optimization of a single objective model with target constraints on all but one objective finds a noninferior, or Pareto optimal, solution; and 2) for some classes of problems, noninferior solutions adjacent in objective space map to adjacent decision vectors with only marginal differences in the decision space. The attractive features of CMEA include: easily adaptable for use with existing implementation of evolutionary algorithms for an optimization problem since no new operators are needed; and relatively less compute intensive since Pareto optimality is ensured in an implicit manner, and therefore expensive sorting and pair-wise comparison operations that are typically required by other Pareto-based MOEAs are eliminated.

To evaluate the applicability of CMEA to MO problems, it was applied to two test problems with different characteristics and levels of difficulty. Test problems covered continuous as well as combinatorial search, unconstrained as well as constrained optimization, real as well as binary variables, and as few as one variable to as high as 500 variables. This evaluation included performance comparisons with other MOEAs and with mathematical programming-based noninferior solutions. Accuracy, coverage, and spread of the noninferior solutions were used to compare the performance. To evaluate the consistency of CMEA in generating the noninferior set, several random trials were performed. Overall, CMEA performed well with respect to these criteria for both problems tested. The spread and coverage of noninferior solutions obtained using CMEA were always better than those demonstrated by other MOEAs. With respect to accuracy, SPEA did better than CMEA for a narrow range of noninferior solutions, but overall CMEA did better over a broader range of the noninferior set.

Some known limitations of CMEA include the following. The computational efficiency gain obtained in CMEA is predicated on the existence of similarities in noninferior solutions that correspond to adjacent points in the objective space. For problems where this may not hold true strongly, the search implemented by CMEA becomes analogous to solving a number of independent single objective optimization problems, and therefore, may not realize any significant computational gain. For a problem with more than two objectives, incrementally updating the constraint values to obtain an adjacent point is not necessarily as straightforward as is for the two-objective cases presented here. More investigation is needed to evaluate this issue when applying CMEA to higher dimensional problems.

In the present study, the number of functions evaluations was used as a measure to compare the computational needs of CMEA with that of a single objective-based MO analysis without seeding. CMEA required approximately 75% fewer function evaluations. This measure alone is not sufficient to compare the computational gain, if any, that may be realized by CMEA over the other MOEAs that use explicit Pareto optimality ordering. The computational performance of CMEA and other MOEAs needs to be studied further.

References

1. Chetan, S. K., (2000). *Noninferior Surface Tracing Evolutionary Algorithm (NSTEA) for Multiobjective Optimization*, MS Thesis, North Carolina State University, Raleigh, North Carolina.
2. Cieniawski, S.E., Eheart, J. W., and Ranjithan, S., (1995). Using Genetic Algorithms to Solve a Multi-Objective Groundwater Monitoring Problem, *Water Resources Research*, vol. 31, no. 2, pp. 399-409.
3. Coello, C.A.C., Christiansen, A.D., and Aguirre, A.H., (1998). Using a New GA-Based Multiobjective Optimization Technique for the Design of Robot Arms, *Robotica*, 16(4), pp. 401-414
4. Coello, C. A. C., (1999a). A comprehensive survey of evolutionary-based multiobjective optimization techniques, *Knowledge and Information System*, 1(3), pp. 269-308.
5. Coello, C. A. C., (1999b). *List of references on evolutionary multiobjective optimization.* Available: www.lania.mx/~ccoello/EMOO
6. Coello, C.A.C. and Christiansen, A.D., (2000). Multiobjective optimization of trusses using genetic algorithms, *Computers and Structures*, 75(6), pp. 647-660.
7. Cohon, J.L., (1978). *Multiobjective programming and planning*, Mathematics in Science and Engineering, Vol. 140, Academic Press, Inc.
8. Fonesca, C.M., and Fleming, P.J., (1993). Genetic Algorithms for multiobjective optimization: Formulation, Discussion and generalization*, Genetic Algorithms: Proceedings of Fifth International Conference*, pp. 416-423.
9. Fonesca, C.M, and Fleming, P.J., (1995). An overview of evolutionary algorithms in multiobjective optimization, *Evolutionary Computation*, 3(1), pp. 1-16.
10. Hajela, P. and Lin, C.-Y.,(1992). Genetic search strategies in multicriterion optimal design, *Structural Optimization*, 4, pp. 99-107.
11. Harrell, L. J. and Ranjithan, S., (1997). Generating Efficient Watershed Management Strategies Using a Genetic Algorithms-Based Method, Ed: D. H. Merritt *Proceedings of the 24th Annual Water Resources Planning and Management Conference (ASCE)*, Houston, TX, April 6-9, 1997, pp. 272-277.
12. Horn, J., and Nafpliotis, N. and Goldberg, D.E., (1994). A niched Pareto genetic algorithm for multiobjective optimization, *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 1, pp. 82-87.
13. Horn, J., (1997). *Multicriterion decision making*, In Back, T., Fogel, D., and Michalewicz, Z., editors, Handbook of Evolutionary Computation, Volume 1, pp. F1.9:1-F.1.9:15, Oxford University Press, Oxford, England.

14. Jiménez, J. and Cadenas, J.M., (1995). An evolutionary program for the multiobjective solid transportation problem with fuzzy goals, *Operations Research and Decisions*, 2, pp. 5-20.
15. Knowles, J.D., and Corne, D.W., (2000). Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy, *Evolutionary Computation*, 8(2): pp. 149-172.
16. Loughlin, D.H., and Ranjithan, S., (1997). The neighborhood constraint method: a genetic algorithm-based multiobjective optimization technique, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pp. 666-673.
17. Loughlin, D. H., Ranjithan, S., Brill, E. D., and Baugh, J. W., (2000a). Genetic algorithm approaches for addressing unmodeled objectives in optimization problems, to appear in *Engineering Optimization (in print).*
18. Loughlin, D. H., Ranjithan, S., Baugh, J. W., and Brill, E. D., (2000b). Application of Genetic Algorithms for the Design of Ozone Control Strategies, *Journal of the Air and Waste Management Association,* 50, June 2000, pp. 1050-1063.
19. Menczer, F, Degeratu, M., and Street, W. N., (2000). Efficient and scalable Pareto optimization by evolutionary local selection algorithms, *Evolutionary Computation*, 8(2), pp. 223-247.
20. Obayashi, S., Sasaki, D., and Hirose, N., (2000). Multiobjective Evolutionary Computation for Supersonic Wing-Shape Optimization, *IEEE Transactions on Evolutionary Computation,* 4(2), pp. 182-187.
21. Ritzel, B. J., Eheart, J. W., and Ranjithan, S., (1994). Using Genetic Algorithms to Solve a Multi-Objective Groundwater Remediation Problem, *Water Resources Research*, vol. 30, no. 5, pp. 1589-1603.
22. Schaffer, J.D., (1984). Multiple objective optimization with vector evaluated genetic algorithms, Ph.D. Thesis, Vanderbilt University.
23. Schaffer, J.D., (1985). Multiple objective optimization with vector evaluated genetic algorithms, Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms, pp. 93-100.
24. Srinivas, N., and Deb, K., (1994). Multiobjective optimization using nondominated sorting in genetic algorithms, *Evolutionary Computation*, 2(3), pp. 221-248.
25. Van Veldhuizen, D.A., and Lamont, G.B., (2000). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art, *Evolutionary Computation*, 8(2), pp. 125-147.
26. Zitzler, E., and Thiele, L., (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach, *IEEE Transactions on Evolutionary Computation*, 3(4), pp. 257-271.
27. Zitzler, E., Deb, K., and Thiele, L., (2000). Comparison of multiobjective evolutionary algorithms: empirical results, *Evolutionary Computation*, 8(2), pp. 173-195.